1. A method of implementing an atomic transaction using a program logic, said method comprising:

requesting in said program logic a transaction identifier for said atomic transaction;

generating said transaction identifier in a transaction manager in response to said requesting;

specifying in said program logic a plurality of combinations for execution in a sequential order, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a roll-back procedure, wherein said task procedure implements a part of said atomic transaction and said roll-back procedure is designed to roll-back said task procedure;

executing said task procedures in said sequential order;

keeping track of said roll-back procedures in said transaction manager; and

executing said roll-back procedures in a reverse order of said sequential order if said atomic transaction is to be aborted in the middle, wherein said roll-back procedures are identified according to said keeping.

[c2]

2. The method of claim 1, wherein said transaction identifier is unique to each of the atomic transactions.

[c3]

3. The method of claim 1, wherein said keeping comprises storing data representing said roll-back procedures in a stack.

[c4]

4. A computer readable medium carrying one or more sequences of instructions representing a program logic for execution on a system, said program logic implementing an atomic transaction, wherein execution of said one or more sequences of instructions by one or more processors contained in said system causes said one or more processors to perform the actions of:

requesting an identifier for said atomic transaction;

setting a variable to equal said identifier;

specifying a plurality of combinations for execution, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a roll-back procedure, wherein said task procedure implements a part of said atomic transaction and said roll-back procedure is designed to roll-back said task procedure; and

aborting said atomic transaction in the middle of execution by specifying said identifier associated with an abort procedure to cause said roll-back operations to be executed.

[c5]

5. The computer readable medium of claim 4, wherein said specifying comprises including each of said plurality of combinations in a single procedure call.

[c6]

6. A computer readable medium carrying one or more sequences of instructions for supporting implementation of an atomic transaction in a system, wherein execution of said one or more sequences of instructions by one or more processors contained in said system causes said one or more processors to perform the actions of:

generating an identifier for said atomic transaction;

specifying a plurality of combinations for execution, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a roll-back procedure, wherein said task procedure implements a part of said atomic transaction and said roll-back procedure is designed to roll-back said task procedure;

executing said task procedures; and

executing said roll-back procedures in response to receiving an abort request.

[c7]

7. The computer readable medium of claim 6, wherein said task procedures are executed in an execution order and corresponding roll-back procedures are executed in a reverse order of said execution order.

[c8]

8. The computer readable medium of claim 7, further comprising storing data indicating that said roll-back procedures are to be executed in said reverse order to abort said atomic transaction.

[c9]

9. The computer readable medium of claim 8, wherein said identifier is generated to be unique for each atomic transaction.

[c10]

10. The computer readable medium of claim 8, wherein said data is represented in the form of a stack in a memory.

[c11]

11. A computer system comprising:

a memory storing a plurality of instructions; and

a processing unit coupled to said memory and executing said plurality of instructions to support implementation of an atomic transaction in a programming environment, said processing unit being operable to:

request in a program logic a transaction identifier for said atomic transaction;

generate said transaction identifier in a transaction manager in response to said requesting;

specify in said program logic a plurality of combinations for execution in a sequential order, wherein each of said plurality of combinations contains said transaction identifier, a task procedure, and a roll-back procedure, wherein said task procedure implements a part of said atomic transaction and said roll-back procedure is designed to roll-back said task procedure;

execute said task procedures in said sequential order;

keep track of said roll-back procedures in said transaction manager; and

execute said roll-back procedures in a reverse order of said sequential order if said atomic transaction is to be aborted in the middle, wherein said roll-back procedures are identified according to said keeping.

Abstract of Disclosure

[0092]   An aspect of the present invention simplifies the implementation of custom atomic transactions. A program logic (implementing a custom atomic transaction) may request a unique transaction identifier from a programming environment. The program logic may then specify a task procedure, corresponding roll-back procedures, and the transaction identifier using an interface provided by the programming environment. The programming environment keeps track of the specified roll-back procedures. The